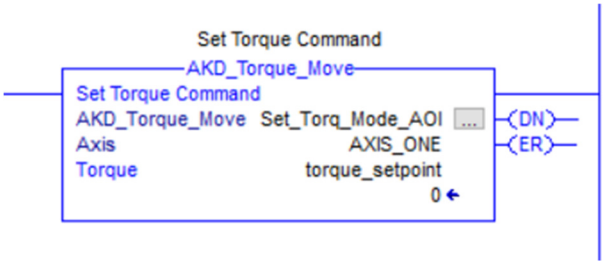
AKD_Torque_Move and using the AKD Ethernet IP drive in Torque Mode

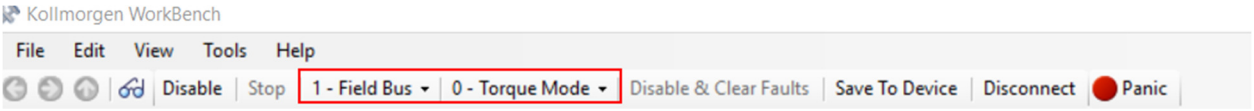
AKD_Torque_Move



Description/Behavior

The AKD_Torque_Move does NOT on execution change the drive command source (DRV.CMDSOURCE) or drive op mode (DRV.OPMODE). It simply sets the command type to 0x05 (torque move) and uses the setpoint to set the IL.CMD directly (it does not use the IL.CMDU as a setpoint). The value used in the AOI for the torque setpoint is such tht XXXX over Ethernet IP equals X.XXX in the AKD. For example, a value in the AOI of 1 is 0.001 or 1mA, 1000 is 1.000 is 1000mA or 1A, etc.

The AKD_Torque_Move on will fail (ER) is the drive is not already in Fieldbus Torque prior to triggering and executing the AOI.

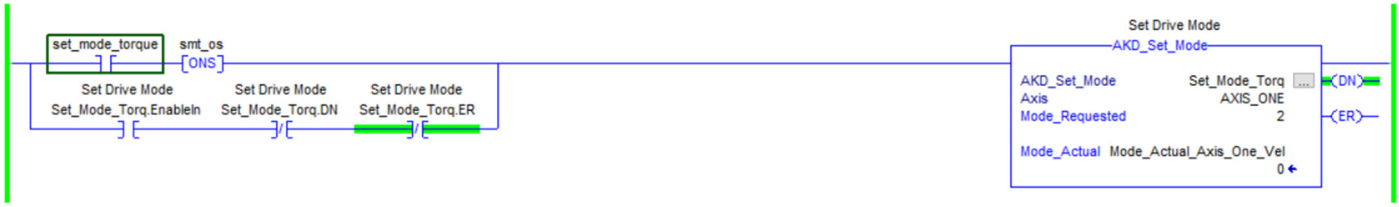


The Torque Move is the only method with the AKD Ethernet IP that uses Fieldbus as a command source.

There are various methods for ensuring the drive is set for the correct command source and op mode before triggering the AKD_Torque_Move AOI.

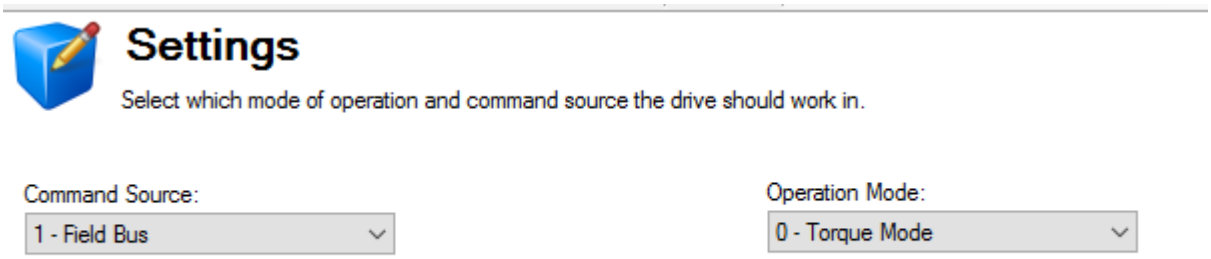
Method#1:

The SET_MODE AOI when the Mode_Requested=2 (torque mode) will set both the DRV.CMDSOURCE=Fieldbus and DRV.OPMODE=Torque.



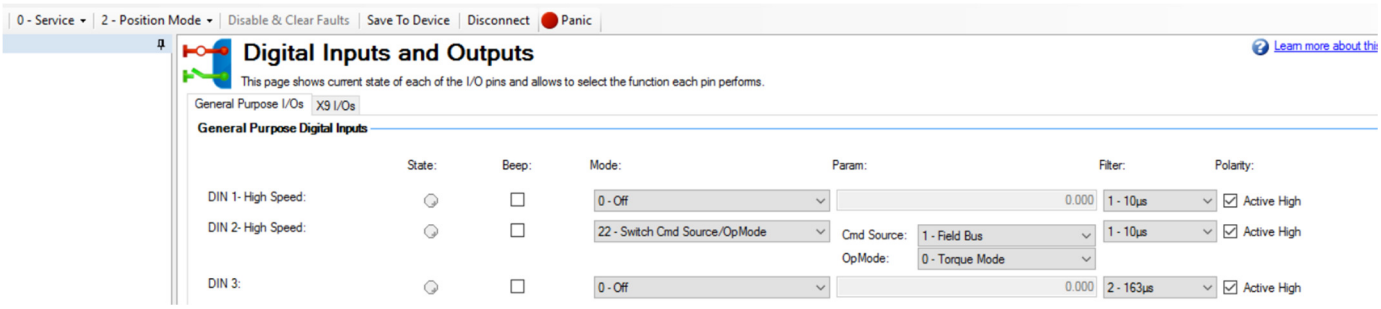
Method#2:

If the drive op mode is always to be torque mode then use Workbench to setup Fieldbus and Torque as the power up command source and op mode.



Method#3:
Digital Inputs can be configured to switch between command source types and op modes but this uses hardware I/O instead of fieldbus to accomplish.

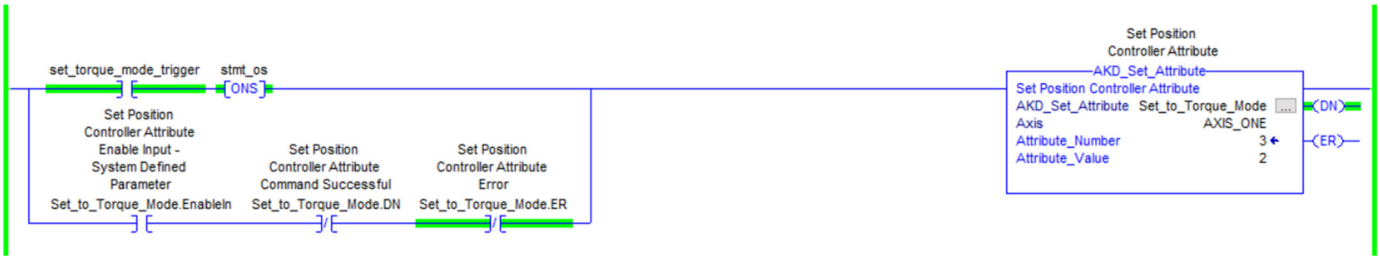
In this example the input toggles between Service Position and Fieldbus Torque. Which command sources and op modes to configure the input for is application dependent.



Method#4: AKD_Set_Attribute AOI

If the set controller attribute is used for set mode and torque mode is requested, both the command source and op mode will change to fieldbus and torque mode respectively.

Here is an example of using the AKD_Set_Attribute to change the DRV.OPMODE to Torque and the DRV.CMDSOURCE to Fieldbus.



The attribute number 3 (Mode) and 2 (Torque Mode) comes from the following table in the EIP manual:

8.1 Position Controller Object 0x25

Attribute ID (Decimal Value)	Name	Access Rule	Type	Description
1	Number of Attrib-utes	Get	USINT	Returns the total number of attributes supported by this object in this device.
2	Attribute List	Get	Array of USINT	Returns an array with a list of the attributes sup-ported by this object in this device.
3	Mode	Get/Set	USINT	Operating mode. 0 = Position mode(default), 1 = Velocity mode, 2 = Torque mode.
4	Position Units	Get/Set	DINT	Position Units ratio value is the number of actual position feedback counts equal to one position unit (default 1).
5	Profile Units	Get/Set	DINT	Profile Units ratio value is the number of actual position feedback counts per second or second2 equal to one velocity, acceleration or deceleration unit (default 1).

Note on Stopping Torque Move:

In the AKD Ethernet IP Communications manual the following is stated.

6.5.2 Torque Moves

Once the drive is setup for torque mode, issue Torque commands (command type 0x05) to set a torque set point in the drive. Torque commands and values are scaled in milliarms.

While in motion, issue another Torque command to immediately change the target torque.

While a torque command is active, the In Motion bit in Status Word 1 will be set and In Position will be cleared. The Direction status bit will reflect the actual direction of motion.

Set the Smooth Stop bit to stop the motor at the previously set deceleration rate and remain enabled.

If the Smooth Stop bit is set directly to the command word byte 0 then the torque move does stop (same behavior as DRV.STOP).

However, if using the AKD_Smooth_Stop AOI, I reported a bug that exists in the production release 1-17-0-0 and prior. The bug was in regards to attempting to stop the Torque Move using the Smooth Stop AOI. “The AKD_Torque_Move add-on-instruction once executed starts a torque move but the AKD_Smooth_Stop add-on-instruction does not stop the move. This is because unlike a Position Move type, the Profile In Progress bit is not set in the firmware during a Torque Move.”

This was fixed in a beta release and will appear in the next production release (date unknown).

Alternative methods would be to disable the drive or set the torque value to zero, etc.

Per the AKD Ethernet IP Communications Manual.

6.2.2.3 Command Type 0x05 - Torque

This command type is used to change the target torque. This can only be used in torque mode. Motion will begin as soon as the value is loaded.

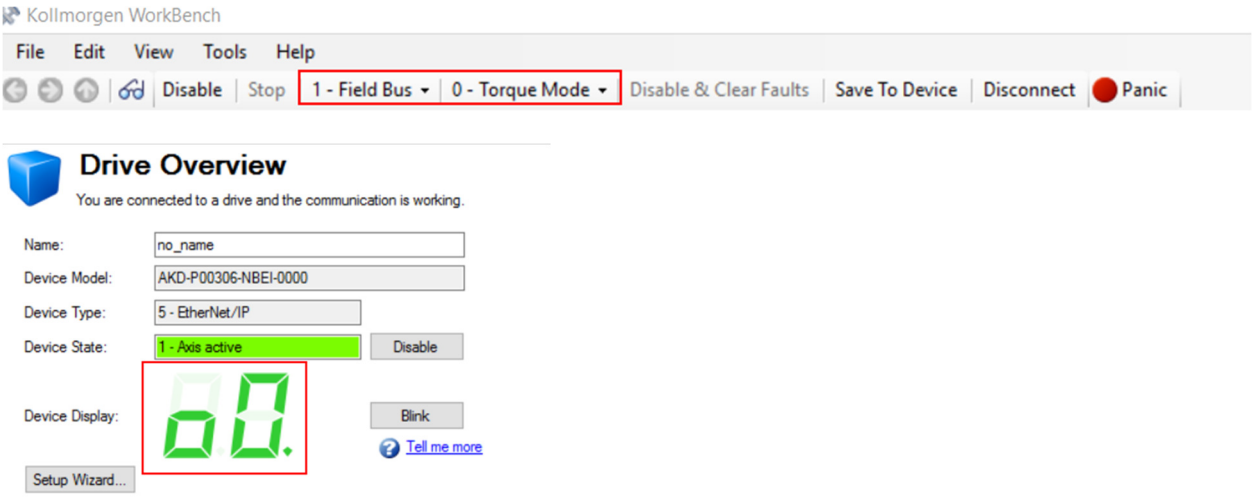
- Put drive in torque mode by sending a message to Position Controller class 0x25, Instance 1, Attribute 3 Operation Mode.
- Load the desired torque value in bytes 4-7.
- Set the Load/Start bit to begin the move.

Torque values are in milliamps [mA].

Making the Torque Move

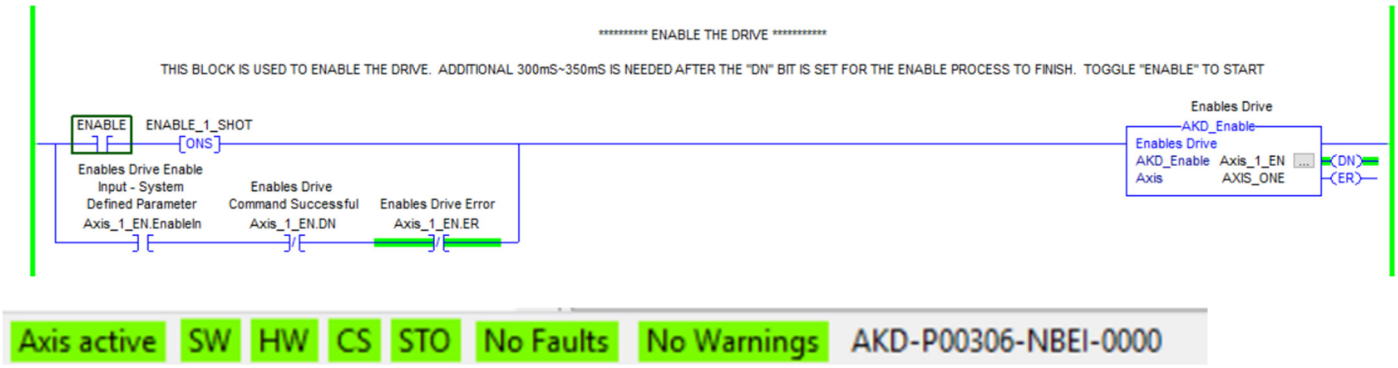
Step 1:

Make sure the drive is in Fieldbus Torque mode using one of the methods presented above.

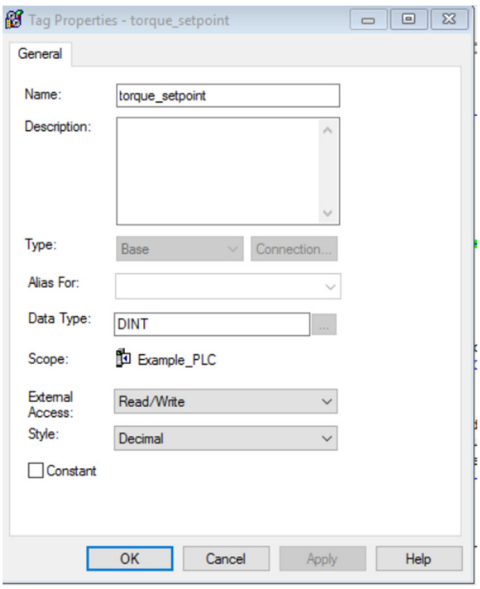


Step 2:

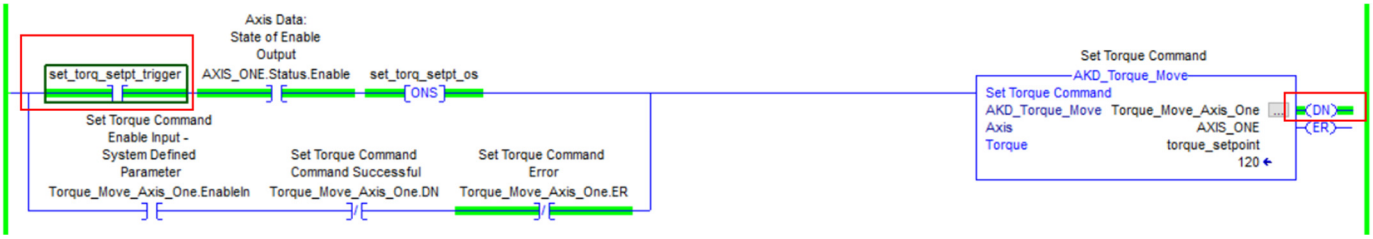
The drive must be enabled.



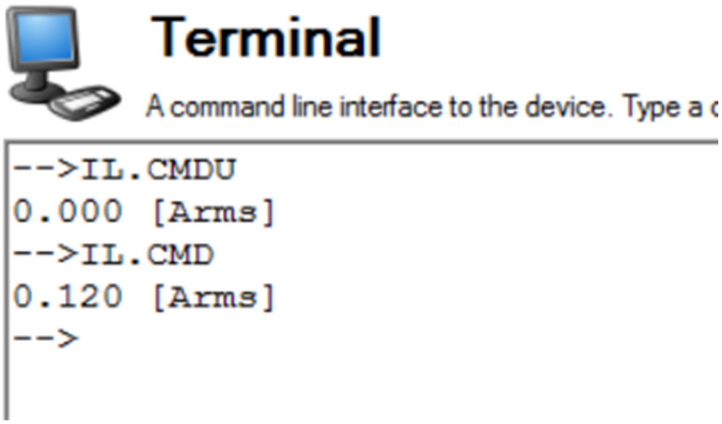
Step 3: Set the AKD_Torque_Move with the desired setpoint value. In this example, I entered a tag into the AKD_Torque_Move block so the value may be varied (as opposed to a hardcoded constant value). The tag is declared as a Data Type: DINT



Step 4: Trigger the Torque Move by toggling the normally open contact which can be done online in RSLogix or based on your PLC program's logic. If successful, the DN bit will turn on.



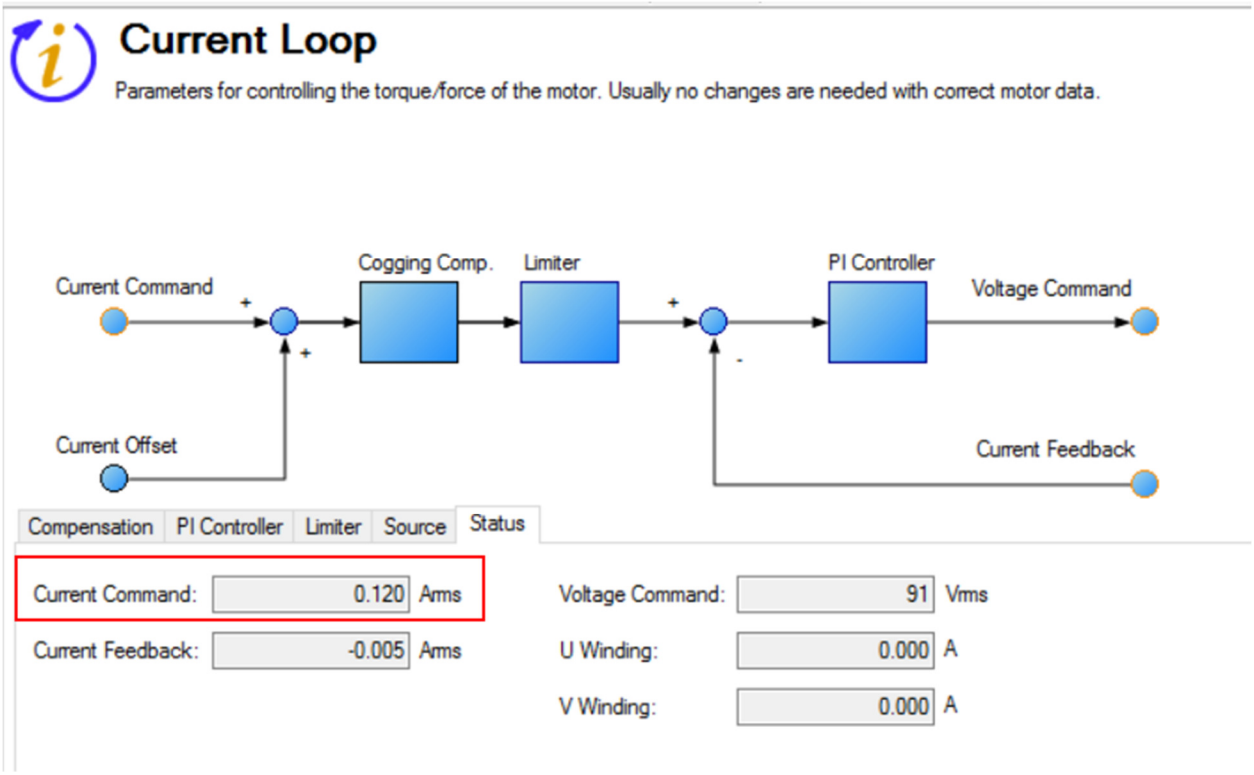
In this case the value in the setpoint of the AOI is 120. This can be queried in Terminal as shown below. This demonstrates the value is passed directly to IL.CMD and IL.CMDU is not used for the Torque Move.



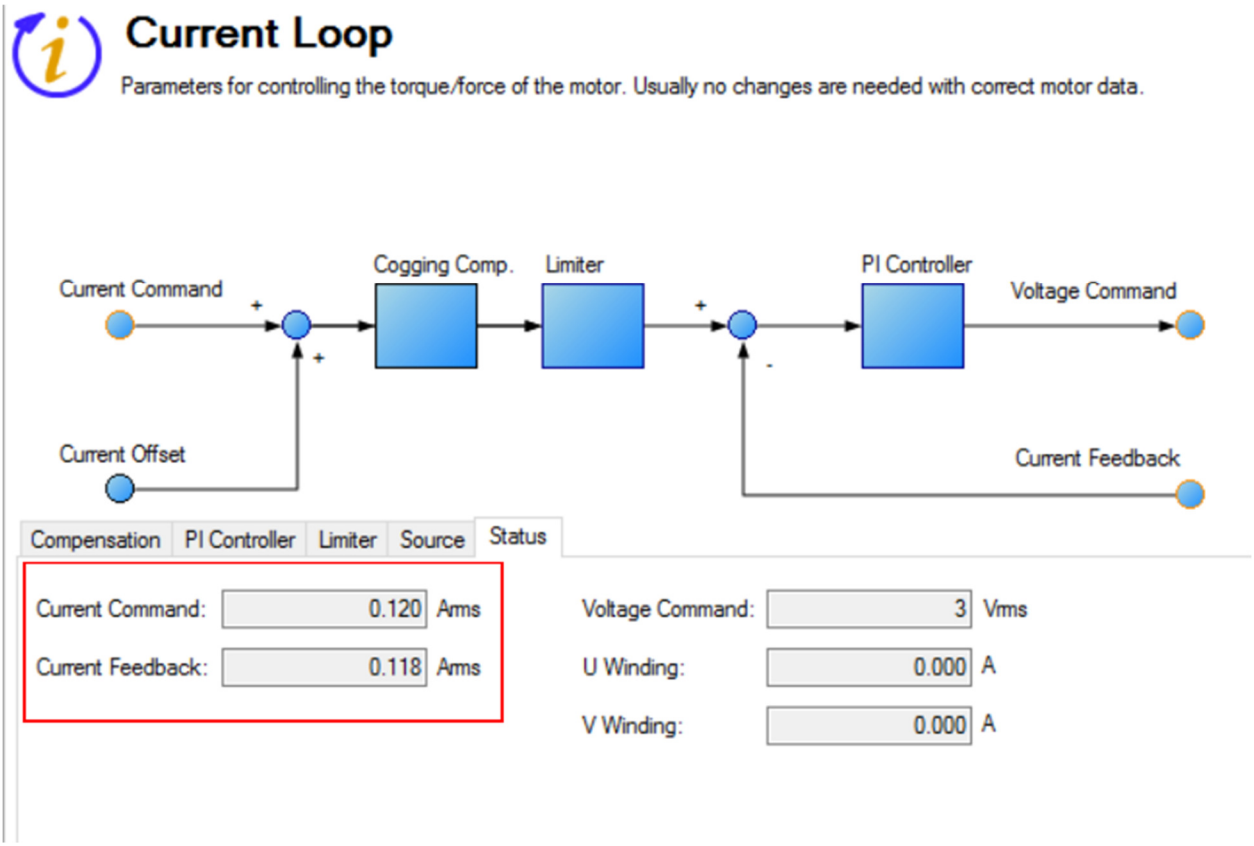
The current can be monitored in other places as well in Workbench or over Ethernet IP.

Workbench

No Load



Loaded



If the torque setpoint needs to change then STEP 3 and 4 must be repeated where the new setpoint is entered and the normally open contact must be toggled from false to true to retrigger the AKD_Torque_Move AOI (in order to update the setpoint command).

Monitoring over EIP

There is an application note on the KDN that demonstrates how to monitor current over EIP:

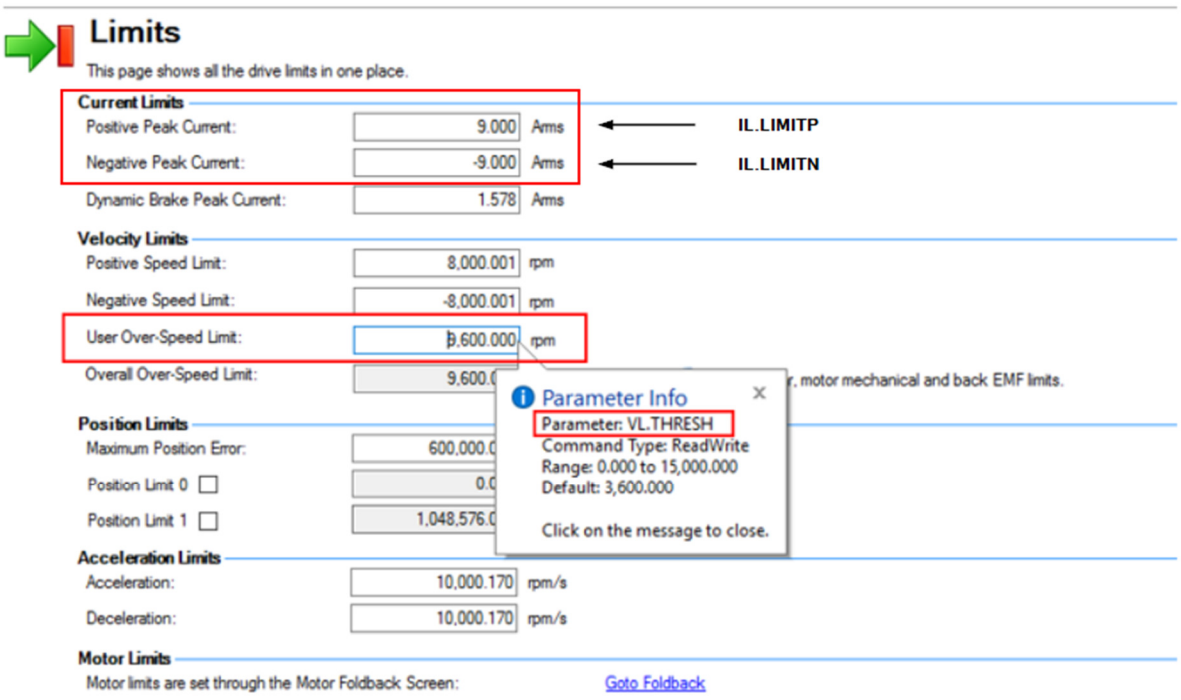
https://www.kollmorgen.com/en-us/developer-network/how-do-i-monitor-current-ilcmd-and-ilfb-over-ethernet-ip/

Additional notes:

Limits

Current Limits

Often in a go to force or torque application, the question comes up how to set/change the drive current limits over Ethernet IP.



The IL.LIMITP and IL.LIMIT N can be written to using different methods. The following describes these different methods for parameter access. Although the article is how to read a drive parameter over Ethernet IP, the method of writing to a parameter will be similar.

https://www.kollmorgen.com/en-us/developer-network/reading-drive-parameter-akd-ethernet-ip/

Speed Limits

The AKD torque drive op mode is a pure torque controller (current loop regulator). There is no line speed over-ride mechanism where the drive switches to velocity mode under an unloaded condition.

However, there is a parameter called IL.VLIMIT that sets a velocity limit in torque mode. The method is to reduce the current if necessary to keep the VL.FB at or below the IL.VLIMIT.

Be aware that if the value is set too low then it will affect the current loop such it will reduce the current to maintain the velocity limit but you won't have enough headroom for the current loop to regulate about your torque setpoint. I've seen attempts to set the IL.VLIMIT to 1 rpm or less for example.

Description

IL.VLIMIT sets the velocity limit for operation in torque mode.

When set to a non-zero value, the current supplied to the motor will be limited to prevent the drive from accelerating beyond -IL.VLIMIT / IL.VLIMIT.

Related Topics

Current Loop

I queried the velocity feedback in terminal with IL.VLIMIT set to 0 (default) and then set the IL.VLIMIT to 1000 (your velocity units is set to 65536 (EIP default) where 65536=1 rps or 60 rpm if you are following best practices and your Workbench units are set to the same as EIP). I set mine to RPM for demonstrational purposes:

```
-->VL.FB
3021.375 [rpm]
-->IL.VLIMIT
0.000 [rpm]
-->IL.VLIMIT 1000
-->VL.FB
1034.172 [rpm]
-->VL.FB
989.447 [rpm]
-->VL.FB
1032.518 [rpm]
-->VL.FB
1017.087 [rpm]
-->
```

Note I also reported a bug where someone was attempting to use torque mode but also set the IL.OFFSET in the current loop to a non-zero value.

The IL.OFFSET influenced the velocity limit (i.e. proportional to the offset plus the IL.VLIMIT value). The change will be that IL.OFFSET does not affect the limit as set by the IL.VLIMIT.

If your IL.OFFSET is zero then this bug will not affect you, otherwise it will be in the next beta release.

Torque or Force Applications that require accuracy

A few remaining notes is to keep in mind that your data sampling will be affected by the RPI scan setting at least and possibly PLC scan time as well.

We are not familiar with methods or techniques in Allen Bradley as far as data acquisition or recording.

Also note a FAQ that comes up often is customers who wants to convert amps to torque or force mathematically (there is no torque or torque percent parameter in the AKD; only amps).

If you are using AKM motors for example the Kt constant (i.e. N*m/Arms) we publish is +/-10% from motor to motor even if they have identical part#s. This is a variance intrinsic to the motor manufacturing, etc.

Also note losses that are present in the power transmission of every system such as belt compliance, backlash of teeth, friction, etc. that will affect the accuracy of your data since it is based only on motor load (amps).

The only way to truly determine or measure the torque or force applied at the load with accuracy is via torque or force transducers and to use data acquisition hardware/software to record the transducers' information.